Radiology: Artificial Intelligence

EDITORIAL

Magician's Corner: 5. Generative Adversarial Networks

Bradley J. Erickson, MD, PhD • Jason Cai, MD

From the Department of Radiology, Mayo Clinic, 200 First St SW, Rochester, MN 55905. Received November 29, 2019; revision requested December 31; revision received January 14, 2020; accepted January 22. **Address correspondence to** B.J.E. (e-mail: *bje@mayo.edu*).

Conflicts of interest are listed at the end of this article.

Radiology: Artificial Intelligence 2020; 2(2):e190215 • **https://doi.org/10.1148/ryai.2020190215** • **Content code:** IN • ©RSNA, 2020

"Everything you can imagine is real."
Pablo Picasso

Generative adversarial networks (GANs) are a fascinating new technology first described by Goodfellow et al in 2014 (1). Compared with most deep learning applications which are focused on classifying an image (2) or segmenting structures within an image (3), GANs are focused on the task of *creating* images. It is the technology behind DeepFakes (4), where images are generated that look completely real. For instance, the website *http://ThisPersonDoesNotExist.com* repeatedly generates images of humanlike faces, but there is no human used to create the photograph, or at least not directly. GANs also can be used for "style transfer," where a given image is transformed into a different style (5). For example, one may take any photograph and apply a style like van Gogh's paintings to that photograph. Figure 1 shows an example of a photograph I took and then applied the van Gogh style to it, using the styleGan transfer routine. Finally, GANs also can create "superresolution" images (6) from lower-resolution images—essentially creating detail out of nothing.

Although the task of a radiologist is mostly to interpret images of real life, it is becoming apparent that GANs will have a substantial impact on the way we practice radiology in the not-too-distant future. Each of the tasks described above could impact how we practice radiology. The first task, to create images from nothing, would seem dangerous, and indeed it would be if totally unconstrained. However, if we are able to constrain it to some reasonable degree, it may be possible to create much larger training sets than could be feasibly obtained from existing databases. It has been shown that GAN-based generation of datasets results in better training of both traditional machine learning and deep learning systems than traditional image augmentation of real data for both medical and nonmedical tasks (7).

Many articles have been published on the use of GANs to "translate" images from one modality to another: for instance, to convert MR images to CT images used to create attenuation-correction maps for PET/MR scanners (8). GANs have been developed to convert CT images to PET images (9), create MR images from other contrast types of MRI, and reduce radiation dose or contrast material dose (10,11).

This article introduces how GANs work, provides some simple one-dimensional (1D) examples that the user can experiment with and then moves on to produce two-dimensional (2D) MR images of the head.

There are three critical components of a GAN (see Fig 2): *(a)* a collection of real-world examples that the GAN is tasked with simulating; *(b)* a generator that seeks to create artificial examples that look "realistic;" and *(c)* a discriminator that determines if an example presented to it is real (that is, from the collection of real images) or fake (from the generator). The approach is called an *adversarial network* because the generator and the discriminator are in a contest—each tries to beat the other. At the start, the discriminator has never seen an image, and so essentially it will be guessing randomly. Similarly, our generator has not seen an image before, so initially it will create something that looks nothing like an image. Both components receive feedback about whether they are right, so the discriminator will start to learn what real images look like, and the generator will also start to learn which examples it produces are more real.

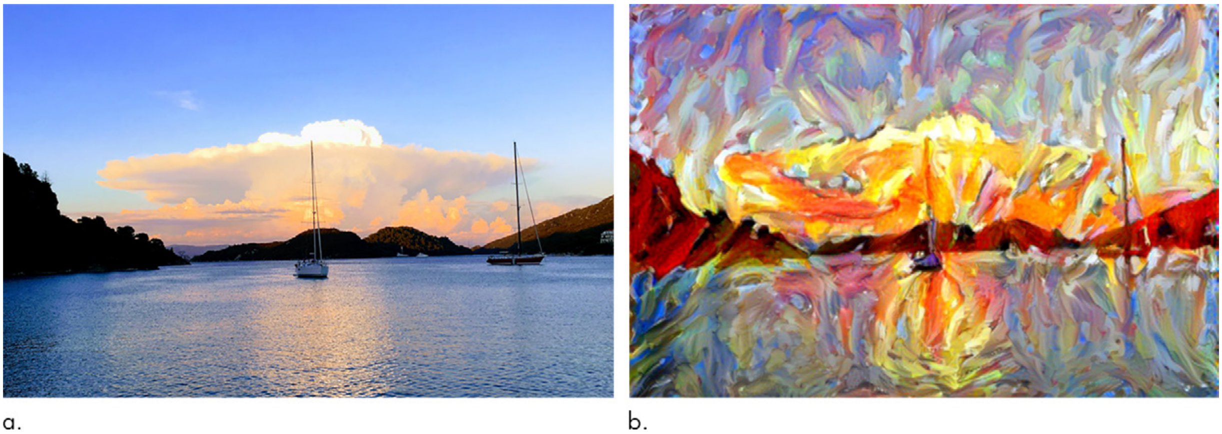

Figure 1: Style transfer using a generative adversarial network. In this case, **(a)** a routine photograph has the **(b)** style of van Gogh applied to it, resulting in an image that has a style similar to how van Gogh might have painted such a scene.

Let's now investigate these parts more closely. The discriminator is essentially a classifier similar to what we built in the first two articles in this series (2,12). It receives both real and fake images and gets feedback about when it correctly predicts the class. The generator has a more challenging task: it must create an image from scratch that is "like" the real images. But what type of images? A picture of a dog, or a cat, or a boat or a snowstorm or an MR image of the head or a CT of the chest, or …? The number of degrees of freedom is huge, and training the generator is a much greater challenge than training the discriminator. Because both must be trained on these challenging tasks, training a GAN tends to take much longer than other artificial intelligence (AI) tasks.

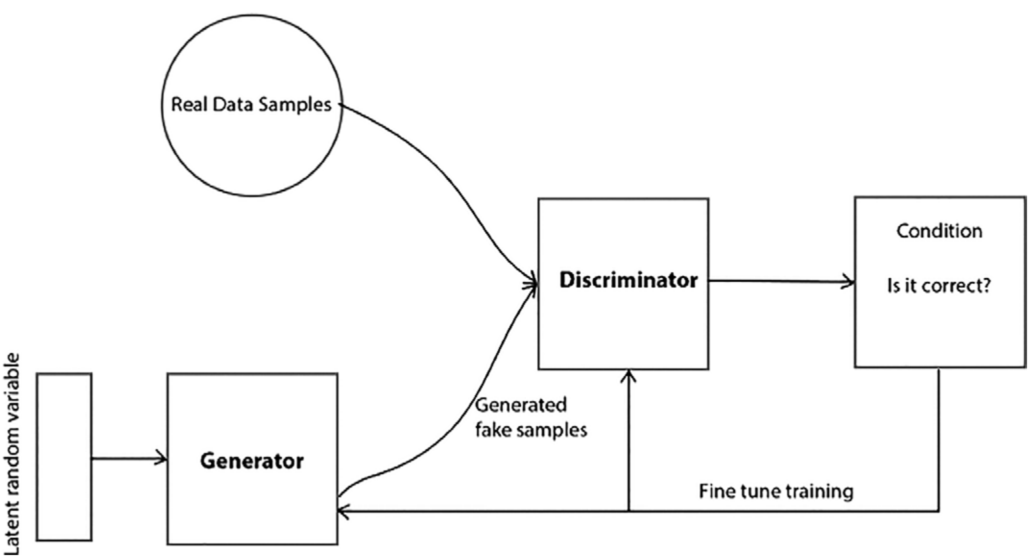

Figure 2: The architecture of a generative adversarial network consists of a set of real images, a generator that learns to create images similar to the real ones, some source of noise to generate variability, and a discriminator that learns to tell the difference between real images and fake ones.

To keep computation times reasonable, we will begin by working on 1D signals rather than an image, which is a 2D signal. We will create 1D patterns that we know and will create a GAN that will learn to generate those patterns. To start, please open the Colab notebook: *http://colab.research.google.com* and select Open Notebook from the File menu, select the Github tab, enter "RSNA" in the search tab, and select the Magicians Corner GAN file.

As before, cell 1 loads the libraries that we will be using (eg, *tensorflow* and *numpy*); please execute that cell. In the first few cells, we create the "real" data generator. In our case, we have five different possible patterns: a sine wave, a square wave, a parabola, a sawtooth, and a circle. The default set up is to cycle through all patterns, but you can create your own pattern, and I would encourage you to do that! Clearly, we could write a function to find these patterns, but of course the key is that we want something that can learn *any* pattern without knowing anything about it beforehand. In our case, our single generator function and the discriminator function will learn any pattern we give it. Also note that it is not just finding the "function," but also the X range. If you change the X or Y range, the GAN will have to start learning again from scratch, but it will learn this new pattern.

In addition to the "real" and "fake" generators, there is a discriminator, which attempts to figure out whether an array of X,Y pairs is "real" (that is, from the "real function" generator) or "fake." Neither the fake generator nor the discriminator is ever allowed to access the real examples—they are given only scores that reflect how well they are doing.

Unlike supervised networks for classification or segmentation, a critical challenge with GANs is that there is no easily computed metric for success. There are two loss function values reported for each epoch: the loss for the discriminator (how well it discriminated real from fake) and the loss for the generator (how many times it fooled the discriminator). These losses are not reliable indicators of generated image quality. At the very start, each is a nearly random guess, so each will get about 50% right. When the generator is creating superb images and the discriminator has learned the nuances to distinguish a really good fake from real, they may well still be at about 50% right. When either the discriminator or generator does better (its loss goes down), the other will get better feedback and thus will learn to improve its performance (and thus the "adversarial" aspect). But this back-and-forth battle can continue forever—each will keep improving, making the other function's score appear worse, even though it likely is doing better with each epoch. If the error rate is very high ($> 90\%$) or very low ($< 10\%$), training will become ineffective, and it may be necessary to restart training (with the hope that a better set of initial conditions will result in a better outcome).

For this reason, it is common practice to have humans observe the output of GANs and cease execution when the results look "good enough." There are limited reports using image quality metrics like Fréchet distance or Fréchet Inception distance, but these have not been highly successful (13).

We are ready to begin creating and executing our GAN. Cell 2 creates the generator and the discriminator functions. It defines the components of the GAN, including functions that retrieve real examples, generate fake examples, and feed those to the discriminator. The function controlling whether real or fake examples are given also provides feedback to both the discriminator and the generator about their respective performance. The number of layers and filters is entirely arbitrary and are not necessarily optimal. Please run cell 2 to create these functions—note that it does not start to train the GAN, it just puts important functions in place.

Cell 3 creates the "real" examples, and several such "real" 1D functions are in this cell. Later, we will create a loop that

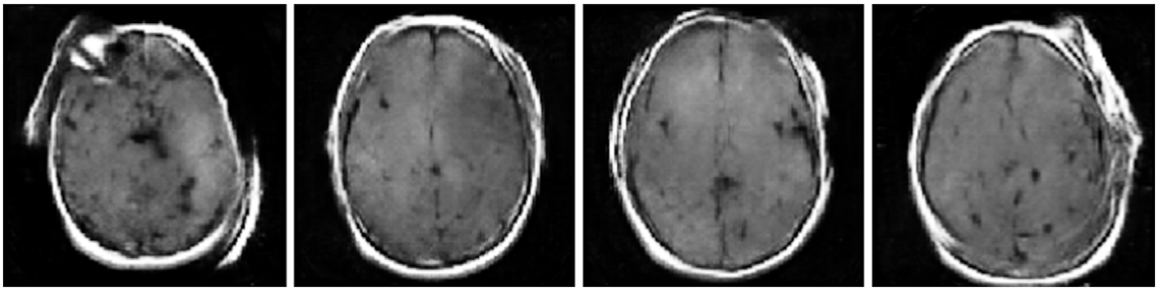

Figure 3: Four example T1-weighted MR images of the brain created by the generative adversarial network (GAN) code in the notebook associated article. There are still clear artifacts associated with these, but given the limited training time, the quality of the images is reasonably good. Typical training times for GANs are hours to days, versus the minutes used for these examples.

exercises all five functions, showing that the same GAN can learn all five patterns. Feel free to create your own pattern and make the GAN learn it. The latter part of cell 3 has some utility functions for calculating the error and also for plotting the function and the GAN generator estimate.

Cell 4 begins the training process for the GAN. The training loop consists of getting a sampling of real points of the 1D pattern ("x_real" and "y_real") and getting the fake examples from the generator. The discriminator is then given both real and fake examples (note the separate training calls with real and fake batches) and begins to learn which is which. Of course we know that we are giving a batch of all real or all fake examples, but the discriminator is blinded to this. The generator is trained (gan_model.train_on_batch) to better approximate the true pattern. Finally, the performance of the discriminator and generator are computed and plotted. The end of the cell loops through the five patterns that we train the GAN to create. With each loop, we show the pattern (red dots) and the estimate created by the GAN (blue dots), at various epochs, allowing you to see the GAN's progress. An important point is that while we know the pattern, neither the generator nor the discriminator does.

On the right-hand side of the graph is a plot of the true error, since we know what the pattern is. In most cases with images, we can't know what the error of the fake image is, since we don't know what truth is. This again reflects the challenge of measuring GAN progress and performance. If either the generator or the discriminator learns faster than the other, the feedback to the other part is almost useless, because essentially every prediction is wrong. When either has partial success, the features useful for either generating or discriminating real versus fake can be amplified, resulting in better GAN performance. This is like the real world: having a 3-year-old child work on multivariate calculus problems won't help them learn anything in math, if all they get is feedback saying they are wrong. They must learn gradually and with feedback at an appropriate level.

Now that you have seen the principles in action, we will create another GAN that actually will create images. Cell 5 loads MR images that we will use as the "real" images. The size of the images is reduced because of computational limits, but feel free to use 256 × 256-pixel images to create higher quality images (at the cost of much slower computation). We also will create a generator and discriminator as before, with the major difference that we do need to encode the fact that now we are working on 2D images, rather than a 1D signal. Please execute cells 6 through 8 to create our Image GAN. The complexity is greater because it is 2D, and so it takes much longer for the system to learn.

Cell 9 contains code to display the images. Cell 10 defines the training function and cell 11 starts the actual training of the image GAN. Please execute these cells. As this GAN learns the MR images, you will see the images created improve from essentially noise at the start to images that are very similar to actual patient images. The initial images have a checkerboard type of pattern because the starting point is random numbers that are approximately replicated using a transpose convolution function, which we used in the previous article (3) for upscaling an image. Figure 3 shows an example T1-weighted image produced by our GAN.

As noted at the start, GANs already are being used in radiology. The ability to create very realistic images can result in better training sets than augmentation methods applied to real images. It would be interesting to understand more about the discriminator and generator themselves, as they must have learned the "essence" that makes an image real. Variants of GANs can create related images, such as making a CT image that matches an MR image. GANs can also produce high-quality images from reduced signal (eg, reduced x-ray dose, reduced radiofrequency signal in the case of MRI, reduced tracer in the case of nuclear medicine, and reduced injected contrast agent for MRI and CT).

As with most powerful technologies, there are important caveats to be aware of. GANs are designed to create realistic images that reflect the training set of real images. In the limiting case, they could potentially be identical to the training images, which is not what is desired. It is also possible to have them create unintended components in an image. For instance, if the training set includes pathologic findings, and if we know that certain findings "belong" with others, but the GAN hasn't learned this, it is possible the GAN will create impossible combinations of findings. There is also concern that a GAN could create pathologic findings when the real images contained no pathologic findings. This reflects the challenge that there is not a clear "training complete" signal that can be used to assure that the GAN will always produce images indistinguishable from the training set. There are concerns that GANs could create pathologic findings or obscure pathologic findings, and that is a legitimate concern if the GAN is not "fully" trained. And since "fully" trained is hard to know, it is difficult to guarantee that GANs will always perform the way we expect.

Some have suggested that AI is at the peak of the "hype cycle," but GANs may be a technology that still has a great deal of untapped potential. LeCun has opined that GANs are the most important advance in machine learning in the past 10 years (14). That statement may be true for radiology AI as well.

Author contributions: Guarantors of integrity of entire study, B.J.E., J.C.; study concepts/study design or data acquisition or data analysis/interpretation, B.J.E., J.C.; manuscript drafting or manuscript revision for important intellectual content, B.J.E., J.C.; approval of final version of submitted manuscript, B.J.E., J.C.; agrees to ensure any questions related to the work are appropriately resolved, B.J.E., J.C.; literature research, B.J.E., J.C.; experimental studies, J.C.; and manuscript editing, B.J.E., J.C.

Disclosures of Conflicts of Interest: **B.J.E.** disclosed no relevant relationships. **J.C.** Activities related to the present article: disclosed no relevant relationships. Activities not related to the present article: employed by Mayo Clinic. Other relationships: disclosed no relevant relationships.

References

1. Goodfellow IJ, Pouget-Abadie J, Mirza M, et al. Generative Adversarial Networks. ArXiv [stat.ML]. [preprint] http://arxiv.org/abs/1406.2661. Posted 2014. Accessed September 2018.
2. Erickson BJ. Magician's corner: How to start learning machine learning. Radiol Artif Intell 2019;1(4):e190072.
3. Erickson BJ, Cai J. Magician's Corner: 4. Image Segmentation with U-Net. Radiol Artif Intell 2020;2(1):e190161.
4. Thies J, Zollhöfer M, Stamminger M, Theobalt C, Nießner M. Demo of Face2Face: real-time face capture and reenactment of RGB videos. ACM SIGGRAPH 2016 Emerging Technologies on - SIGGRAPH '16. 2016.
5. Karras T, Laine S, Aila T. A Style-Based Generator Architecture for Generative Adversarial Networks. ArXiv [cs.NE]. http://arxiv.org/abs/1812.04948. Posted 2018. Accessed May 2019.
6. Ledig C, Theis L, Huszár F, et al. Photo-realistic single image super-resolution using a generative adversarial network. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, July 21–26, 2017. Piscataway, NJ: IEEE, 2017; 105–114.
7. dos Santos Tanaka FHK, Aranha C. Data Augmentation Using GANs. ArXiv [cs.LG]. http://arxiv.org/abs/1904.09135. Posted 2019. Accessed September 2019.
8. Huo Y, Xu Z, Bao S, Assad A, Abramson RG, Landman BA. Adversarial synthesis learning enables segmentation without target modality ground truth. 2018 IEEE 15th International Symposium on Biomedical Imaging (ISBI 2018), 2018; 1217–1220.
9. Ben-Cohen A, Klang E, Raskin SP, et al. Cross-modality synthesis from CT to PET using FCN and GAN networks for improved automated lesion detection. Eng Appl Artif Intell 2019;78:186–194.
10. Gong E, Pauly JM, Wintermark M, Zaharchuk G. Deep learning enables reduced gadolinium dose for contrast-enhanced brain MRI. J Magn Reson Imaging 2018;48(2):330–340.
11. Wolterink JM, Leiner T, Viergever MA, Išgum I. Generative Adversarial Networks for Noise Reduction in Low-Dose CT. IEEE Trans Med Imaging 2017;36(12):2536–2545.
12. Erickson BJ. Magician's Corner: 2. Optimizing a Simple Image Classifier. Radiol Artif Intell 2019;1(5):e190113.
13. Shmelkov K, Schmid C, Alahari K. How good is my GAN? Proceedings of the European Conference on Computer Vision (ECCV), 2018; 213–229.
14. What are some recent and potentially upcoming breakthroughs in deep learning? Quora. https://www.quora.com/What-are-some-recent-and-potentially-upcoming-breakthroughs-in-deep-learning. Accessed November 29, 2019.